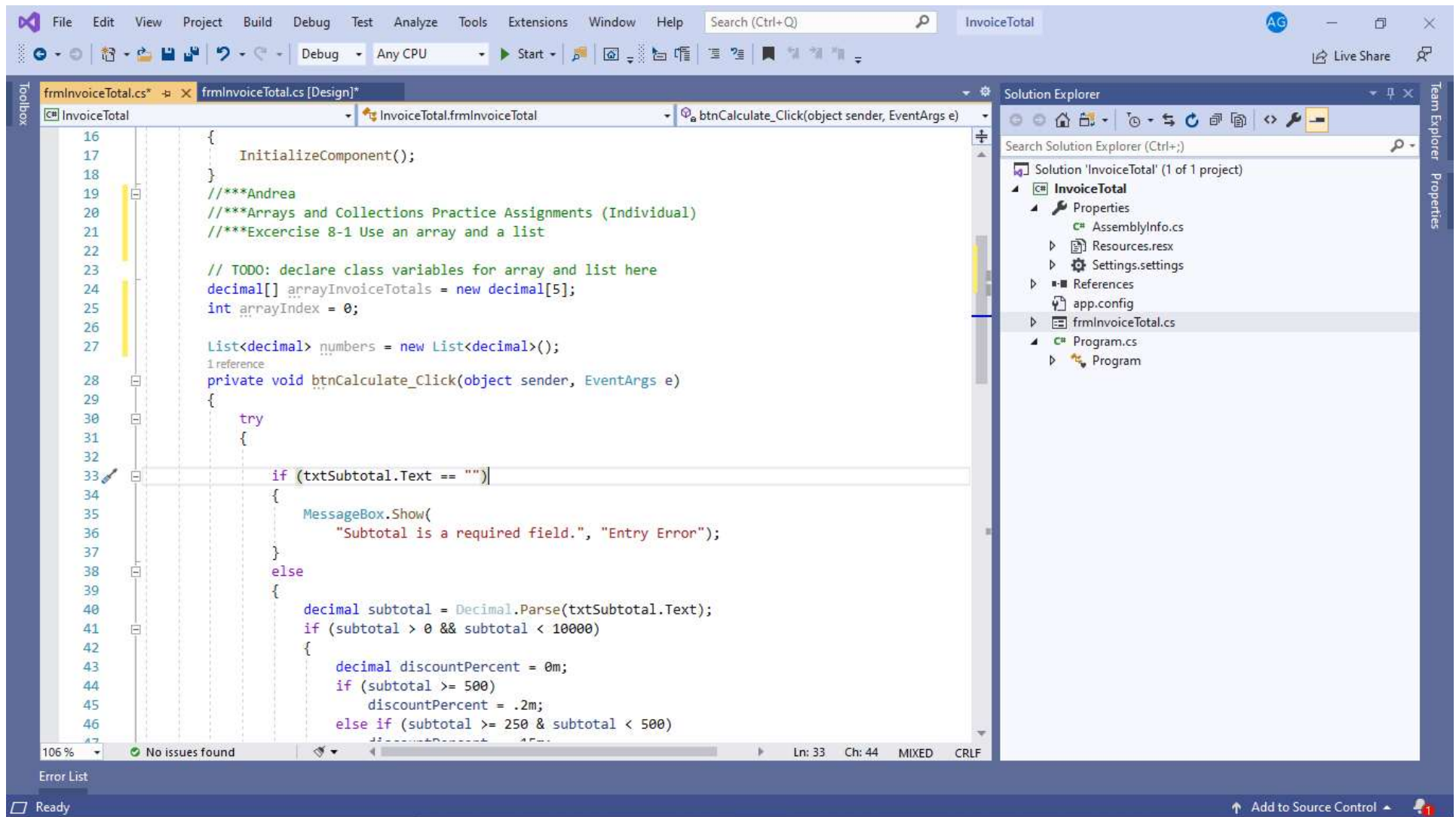


## Assignment: Arrays and Collections Practice Assignments (Individual)

### Exercise 8-1 Use an array and a List #'s 1-10



The screenshot shows the Visual Studio IDE with the file `frmInvoiceTotal.cs` open in Design view. The code is for a Windows Form application named `InvoiceTotal`. The `InitializeComponent()` method is shown at the top. Below it, there are comments indicating the assignment: `/**Andrea`, `/**Arrays and Collections Practice Assignments (Individual)`, and `/**Exercise 8-1 Use an array and a list`. A TODO comment states: `// TODO: declare class variables for array and list here`. The code then declares a `decimal[] arrayInvoiceTotals = new decimal[5];` and an `int arrayIndex = 0;`. A `List<decimal> numbers = new List<decimal>();` is also declared. The `btnCalculate_Click` event handler is implemented with a try-catch block. Inside the try block, there is an if statement: `if (txtSubtotal.Text == "")`. If true, it shows a message box: `MessageBox.Show("Subtotal is a required field.", "Entry Error");`. Otherwise, it parses the subtotal: `decimal subtotal = Decimal.Parse(txtSubtotal.Text);`. It then checks if the subtotal is greater than 0 and less than 10000. If so, it calculates a discount: `decimal discountPercent = 0m;`. If the subtotal is greater than or equal to 500, the discount is set to 0.2m: `discountPercent = .2m;`. If the subtotal is greater than or equal to 250 and less than 500, the discount is set to 0.15m: `discountPercent = .15m;`. The code is currently at line 33, which is the start of the if statement. The Solution Explorer on the right shows the project structure: `InvoiceTotal` (1 of 1 project) containing `AssemblyInfo.cs`, `Resources.resx`, `Settings.settings`, `References`, `app.config`, `frmInvoiceTotal.cs`, `Program.cs`, and `Program`. The status bar at the bottom indicates 106% zoom, no issues found, and the current line is 33, column 44.

```
16 {
17     InitializeComponent();
18 }
19 /**Andrea
20 /**Arrays and Collections Practice Assignments (Individual)
21 /**Exercise 8-1 Use an array and a list
22
23 // TODO: declare class variables for array and list here
24 decimal[] arrayInvoiceTotals = new decimal[5];
25 int arrayIndex = 0;
26
27 List<decimal> numbers = new List<decimal>();
28 private void btnCalculate_Click(object sender, EventArgs e)
29 {
30     try
31     {
32
33         if (txtSubtotal.Text == "")
34         {
35             MessageBox.Show(
36                 "Subtotal is a required field.", "Entry Error");
37         }
38         else
39         {
40             decimal subtotal = Decimal.Parse(txtSubtotal.Text);
41             if (subtotal > 0 && subtotal < 10000)
42             {
43                 decimal discountPercent = 0m;
44                 if (subtotal >= 500)
45                     discountPercent = .2m;
46                 else if (subtotal >= 250 & subtotal < 500)
47                     discountPercent = .15m;
```

Visual Studio interface showing the code editor for `frmInvoiceTotal.cs` in the `InvoiceTotal` project. The code is for the `btnCalculate_Click` event handler.

```
52
53     discountAmount = Math.Round(discountAmount, 2);
54     invoiceTotal = Math.Round(invoiceTotal, 2);
55
56     txtDiscountPercent.Text = discountPercent.ToString("p1");
57     txtDiscountAmount.Text = discountAmount.ToString();
58     txtTotal.Text = invoiceTotal.ToString();
59
60     /***Andrea
61     /***Arrays and Collections Practice Assignments (Individual)
62     /***Exercise 8-1 Use an array and a list
63
64     //adds invoice total to next element when user click the calculate button
65     if (arrayIndex < arrayInvoiceTotals.Length)
66     {
67         arrayInvoiceTotals[arrayIndex] = invoiceTotal;
68         arrayIndex++;
69         numbers.Add(invoiceTotal);
70     }
71     else MessageBox.Show("Out of range", "Array Index Error");
72
73     else
74     {
75         MessageBox.Show(
76             "Subtotal must be greater than 0 and less than 10,000.",
77             "Entry Error");
78     }
79
80 }
81 catch (FormatException)
82 {
83     MessageBox.Show(
```

The Solution Explorer on the right shows the project structure:

- Solution 'InvoiceTotal'
- frmInvoiceTotal
- Properties
- Assemblies
- Resources
- Settings
- References
- app.config
- frmInvoiceTotal
- Program

Bottom status bar: 106 % | No issues found | Ln: 72 | Ch: 22 | MIXED | CRLF | Ready | Add to Source Control

Visual Studio interface showing a C# code file named `frmInvoiceTotal.cs` in Design mode. The code implements a `btnExit_Click` event handler.

```
86 }
87 /***Andrea
88 /***Arrays and Collections Practice Assignments (Individual)
89 /***Exercise 8-1 Use an array and a list
1 reference
90 private void btnExit_Click(object sender, EventArgs e)
91 {
92     // TODO: add code that displays dialog boxes here
93
94     decimal sumA = 0;
95     decimal sumList = 0;
96     string numbersAString = "";
97     string numbersListSting = "";
98
99     //for each loop that processes the data in list
100     numbers.Sort();
101     foreach (decimal invoice in numbers)
102     {
103         if (invoice != 0)
104         {
105             numbersListSting += invoice + " \n";
106             sumList += invoice;
107         }
108     }
109     decimal avgList = sumList / numbers.Count();
110     //for each loop that processes the data in array
111     Array.Sort(arrayInvoiceTotals);
112     for (int i = 0; i < arrayInvoiceTotals.Length; i++)
113     {
114         if (arrayInvoiceTotals[i] != 0)
115         {
116             numbersAString += arrayInvoiceTotals[i] + " \n";
117             sumA += arrayInvoiceTotals[i];
118         }
119     }
120 }
```

The Solution Explorer on the right shows the project structure:

- Solution 'InvoiceTotal'
- frmInvoiceTotal
- Properties
- Assemblies
- Resources
- Settings
- References
- app.config
- frmInvoiceTotal
- Program

The status bar at the bottom indicates 106% zoom, no issues found, and the current line is 89, column 51, with MIXED encoding and CRLF line endings.

Visual Studio interface showing the design view of a Windows Form named `frmInvoiceTotal`. The code in the `btnExit_Click` event handler calculates the sum and average of invoice totals from an array.

```
103     numbersListSting += invoice + "\n";
104     sumList += invoice;
105 }
106
107 decimal avgList = sumList / numbers.Count();
108 //for each loop that processes the data in array
109 Array.Sort(arrayInvoiceTotals);
110 for (int i = 0; i < arrayInvoiceTotals.Length; i++)
111 {
112     if (arrayInvoiceTotals[i] != 0)
113     {
114         numbersAString += arrayInvoiceTotals[i] + "\n";
115         sumA += arrayInvoiceTotals[i];
116     }
117 }
118
119 decimal avg = sumA / arrayInvoiceTotals.Length;
120 /***Andrea
121 /***Arrays and Collections Practice Assignments (Individual)
122 /***Exercise 8-1 Use an array and a list
123 //display box for list
124 MessageBox.Show(numbersListSting + "\n" +
125     "Sum List: " + sumList + "\n" +
126     "Average List: " + avgList + "\n", "List Invoice Totals");
127 //display box for array
128 MessageBox.Show(numbersAString + "\n" +
129     "Sum: " + sumA + "\n" +
130     "Average: " + avg + "\n", "Array Invoice Totals");
131
132 this.Close();
133 }
134
```

The Solution Explorer on the right shows the project structure for `InvoiceTotal`, including `frmInvoiceTotal`. The status bar at the bottom indicates 106% zoom, no issues found, and the current position is Line 122, Column 55.



```
for (a in array
```

```
    .Length; i++)
```

```
    Totals[i] + " \n";
```

```
    .s.Length;
```

```
    assignments (Indivi  
    list
```

```
    .n" +
```

```
    a in array
```

```
    .Length; i++)
```

```
    Totals[i] + " \n";
```

```
    .s.Length;
```

```
    assignments (Indivi  
    list
```

```
    .n" +
```

```
    , "List Invoice Totals");
```

Invoice T...

Subtotal: 500

ent: 20.0%

unt: 100.0

400.0

Calculate Exit

List Invoice Totals

212.50  
212.50  
400.0  
400.0  
400.0

Sum List: 1625.00  
Average List: 325.00

OK

Invoice T...

Subtotal: 500

ent: 20.0%

unt: 100.0

400.0

Calculate Exit

Array Invoice Totals

212.50  
212.50  
400.0  
400.0  
400.0

Sum: 1625.00  
Average: 325.00

OK

## Exercise 8-2 Use a rectangular array

The screenshot shows the Visual Studio IDE with the following components:

- Menu Bar:** File, Edit, View, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q)
- Toolbar:** Includes buttons for Run, Stop, Break, and other development actions.
- Project Explorer:** Shows the project structure with 'FutureValue' and 'Form1.cs'.
- Code Editor:** Displays the code for 'Form1.cs'. The code includes a constructor, an 'InitializeComponent()' call, and a 'btnCalculate\_Click' event handler. The event handler calculates the future value based on user input and stores it in a rectangular array.
- Properties Window:** Empty on the right side.
- Team Explorer:** Visible on the far right.
- Status Bar:** Shows '96 %', 'No issues found', and line/column information (Ln: 19, Ch: 35, SPC, CRLF).

```
17 public Form1()
18 {
19     InitializeComponent();
20 }
21 /***Andrea Griffis
22 /*** Arrays and Collections Practice Assignments (Individual)
23 /*** Exercise 8-2 Use a Rectangular Array
24
25 // TODO: Declare the rectangular array and the row index here
26 string[,] arrayFutureValue = new string[10, 4];
27 int arrayIndex = 0;
28
29 private void btnCalculate_Click(object sender, EventArgs e)
30 {
31     try
32     {
33         if (IsValidData())
34         {
35             decimal monthlyInvestment =
36                 Convert.ToDecimal(txtMonthlyInvestment.Text);
37             decimal interestRateYearly =
38                 Convert.ToDecimal(txtInterestRate.Text);
39             int years = Convert.ToInt32(txtYears.Text);
40
41             int months = years * 12;
42             decimal interestRateMonthly = interestRateYearly / 12 / 100;
43
44             decimal futureValue = CalculateFutureValue(
45                 monthlyInvestment, interestRateMonthly, months);
46             txtFutureValue.Text = futureValue.ToString("c");
47             txtMonthlyInvestment.Focus();
48
49             // TODO: Add the calculation to the rectangular array here
50 }
```

Visual Studio interface showing the code editor for `Form1.cs` in the `FutureValue` project. The code implements a calculation for future value based on monthly investment, interest rate, and years.

```
// TODO: Add the calculation to the rectangular array here
if (arrayIndex < arrayFutureValue.GetLength(0))
{
    arrayFutureValue[arrayIndex, 0] = txtMonthlyInvestment.Text;
    arrayFutureValue[arrayIndex, 1] = txtInterestRate.Text;
    arrayFutureValue[arrayIndex, 2] = txtYears.Text;
    arrayFutureValue[arrayIndex, 3] = txtFutureValue.Text;
    arrayIndex++;
}
else MessageBox.Show("Out of range", "Array Index Error");
}
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message + "\n\n" +
        ex.GetType().ToString() + "\n\n" +
        ex.StackTrace, "Exception");
}
}

1 reference
public bool IsValidData()
{
    return
        // Validate the Monthly Investment text box
        IsPresent(txtMonthlyInvestment, "Monthly Investment") &&
        IsDecimal(txtMonthlyInvestment, "Monthly Investment") &&
        IsWithinRange(txtMonthlyInvestment, "Monthly Investment", 1, 1000) &&

        // Validate the Yearly Interest Rate text box
        IsPresent(txtInterestRate, "Yearly Interest Rate") &&
        IsDecimal(txtInterestRate, "Yearly Interest Rate") &&
        IsWithinRange(txtInterestRate, "Yearly Interest Rate", 1, 20) &&
}
```

The status bar indicates 96% code coverage, no issues found, and the cursor is at line 19, column 35. The bottom status bar shows "Ready" and "Add to Source Control".

Visual Studio interface showing the code editor for `Form1.cs` in the `FutureValue` project. The code implements a future value calculation method and a button click event handler.

```
150 {
151     futureValue = (futureValue + monthlyInvestment)
152         * (1 + monthlyInterestRate);
153 }
154
155 return futureValue;
156 }
157
158 // reference
159 private void btnExit_Click(object sender, EventArgs e)
160 {
161     // TODO: Display the rectangular array in a dialog box here
162
163     string FVAString = "";
164     //process array contents
165
166     for (int i = 0; i < arrayFutureValue.GetLength(0); i++)
167     {
168         for (int j = 0; j < arrayFutureValue.GetLength(1); j++)
169             FVAString += arrayFutureValue[i, j] + "\t";
170         FVAString += " \n";
171     }
172     MessageBox.Show(
173         "Inv/Mo. \t" + "Rate \t" + "Years \t" + "Future Value \t" + "\n" +
174         FVAString + "\n",
175         "Future Value Calculations");
176
177     this.Close();
178 }
179
180 }
181 }
```

The Properties window is empty. The status bar shows 96% zoom, no issues found, and the current line is 19, column 35.

Team Explorer Solution Explorer

Ready Add to Source Control



# Future Value Calculations



Inv/Mo.	Rate	Years	Future Value
100	4	10	\$14,774.06
100	4.5	10	\$15,176.51
100	5	10	\$15,592.93

OK